CONNEXIONS™

The Interoperability Report

February 1991 Volume 5, No. 2

ConneXions—
The Interoperability Report tracks current and emerging standards and technologies within the computer and communications industry.

In this issue:

ConneXions is published monthly by Interop, Inc., 480 San Antonio Road, Suite 100, Mountain View, CA 94040, USA. 415-941-3399. Fax: 415-949-1779.

ISSN 0894–5926

From the Editor

Almost two years ago, we started a series of tutorial articles under the heading *Components of OSI.* Originally, I had envisioned this as a fairly short running series, with half a dozen or so articles introducing the structure that makes up the *OSI Reference Model.* However, as I quickly discovered, OSI is an extremely large and ambitious undertaking with many pieces in place (at least architecturally speaking) and even more to come. Therefore, the series of articles has continued, and I cannot promise that it will end soon either (unless of course someone begs me to stop). After the OSI architecture is fully described, there still remains the issue of implementation and deployment, and you should expect to see more articles about pilot projects, experiments, and eventually fully operational OSI networks in future issues of *ConneXions.* Having observed this industry for a number of years as it takes on new technologies and standards, I will not try to make an educated guess as to exactly *when* OSI systems will be as commonplace as say TCP/IP systems are today, but I will promise to report on the developments that lie ahead.

Most of the OSI components we have described so far have "counterparts" in the Internet (TCP/IP) Protocol Suite. FTAM is somewhat similar to FTP, TP4 is somewhat similar to TCP, and so on. This month we look at yet another such candidate, namely VT, or the "Virtual Terminal ASE" which to some extent corresponds to Telnet in the Internet suite. However, VT is more than just a simple way of letting a terminal talk to a remote host while "pretending" to be attached as a local device. Graham Dixon of Churchill College, Cambridge gives an overview of OSI VT and its many applications.

The last few years have seen a great interest in improving congestion conditions in computer networks. Just as the road system, packet switching networks are subject to performance problems under heavy load. Paul McKenney, now with Sequent Computer Systems gives an overview of the research that is ongoing in this area. We also have information about a new DEC Technical Report on Congestion Management.

One of this month's Letters to the Editor concerns the issue of the high-cost of OSI standards documents, and the copyright which exists on this material. It is felt by many that this situation, coupled with the fact that few if any OSI documents are available electronically, is a severe impediment to the progress of OSI acceptance and deployment. We have heard from various sources that the situation may change soon, and will of course keep you posted.

Components of OSI: The Virtual Terminal ASE

by Graham Dixon, Churchill College, Cambridge, England

Introduction

An earlier article in this series [1] has described how the OSI Application Layer is divided into a number of *application service elements* (ASEs), each of which provides a service to some user external to that ASE. There is one user for each of the two end-systems, with the two peer ASEs providing a means of communication between them.

The *Virtual Terminal ASE* (VT) is unique in that it is designed to allow one or both of the *VT-users* to provide a convenient interface with a real human being. To this end it uses a model abstracted from the services provided by real computer terminals. These services are typically those provided by a keyboard and display screen, often supplemented by an audible alarm or other signals.

Most of the development that has taken place with VT has considered a human user of a terminal end-system interacting with an application package in the other end-system. In this situation, VT may allow the application package to download a substantial amount of local processing ability to the terminal end-system. However, VT may also be used to provide a symmetric communication link between two human users or two application packages. It is also possible for the user at either end to be another ASE.

VT Service and Protocol

In principle the Virtual Terminal ASE permits a number of different classes of service. At the time of writing only the *Basic Class* has been developed, which has been abstracted from character-oriented terminals. Future development could for example define a Graphics Class abstracted from graphics terminals, but such developments are outside the scope of this article. The service provided by Basic Class VT to the VT-user is specified in ISO 9040 [2]. This service includes a number of optional *functional units* that may be invoked independently, but for simplicity this article will describe the total available service without reference to these functional units.

The information that is input to or output from each VT-user must be communicated to the peer VT-user. The protocol for this purpose is specified in ISO 9041 [3], which both defines an *abstract syntax* for the description of items of VT information and gives the action to be taken on receipt of each such item from the peer. The facilities for transmission to the peer VT-user are provided by the Presentation Layer, as described elsewhere in this series [4].

VTE-profiles

The two International Standards ISO 9040 and 9041 are the base standards of the Virtual Terminal ASE. However, they do not in themselves provide a full implementable specification. They recognise that there is such a wide range of uses for character-oriented terminals that it would be unreasonable to attempt one universal specification for all purposes. These base standards thus introduce the concept of a *Virtual Terminal Environment* (VTE) which determines the facilities currently available and which is described by a large number of *VTE-parameters.*

The number of available VTE-parameters is sufficiently large that it would be impracticable to negotiate directly the values to be used for each of them. Instead, a predefined VTE is chosen from a number of *VTE-profiles* that are in some ways analogous to the different generic types that exist of real terminals. However, a VT implementation may include a number of different VTE-profiles.

Negotiation of the VTE-profile to be used is part of the process of setting up the *VT-association* between the two VT ASEs that wish to communicate. A mechanism also exists for switching between VTE-profiles during the tenure of a particular VT-association.

The base standards require the establishment of an *ISO Registration Authority* for the registration of VTE-profile definitions, but at the time of writing this has not yet been established. In the meantime the three *Regional Workshops* NIST OIW, EWOS and AOW are cooperating with one another in the development of such definitions and each is providing its own registration procedure for the VTE-profiles for which it has primary responsibility. The NIST OIW VTE-profiles are published regularly in the NIST *Stable Implementation Agreements* [5], and these Regional Workshops are described in an earlier article in this series [6].

International Standardized Profiles

Two real terminals of the same generic type may still differ, for example between monochrome and colour or in the number of lines of characters that may be displayed on the screen. Such differences also have their counterpart in VT. VTE-profiles may leave some VTE-parameters to be negotiated at the time that the VTE-profile is selected for use.

One of the intentions of VT standardization was to reduce substantially the number of different terminal types that an application package may experience. The functionality of distinct real terminals would be mapped through the Virtual Terminal ASE on to a single virtual terminal. The application package would then deal only with this single virtual terminal and the diversity of real terminals would be hidden from it. It is clear from the above that this has not happened, and that there is scope for as much diversity among virtual terminals as exists in the real world among real terminals. Were such a diversity actually realized, much of the potential benefit of VT would have been lost.

The final part of the standardization process as it affects VT is aimed at preventing such proliferation. The VT base standards will be supplemented by a small number of *International Standardized Profiles* (ISPs). ISPs are a new class of documents recently introduced by ISO, as described in [6]. Each ISP will specify a single VTE-profile together with constraints on the availability and use of values for its parameters. These constraints will be such that a terminal implementation and an application package will always be able to interwork if they are conformant to the same ISP.

Display objects

The screen of a real terminal is modelled in VT by a *display object* (DO). A display object is a one, two or three-dimensional array of *array elements* together with certain auxiliary structures. The dimensions are labelled as x, y and z. For a particular DO, each dimension may be either bounded or unbounded, and each is addressed by an integer coordinate with a minimum value of 1. One of the auxiliary structures is the *display pointer* which holds a set of coordinates and which is modelled on the cursor of a real terminal. It is permitted for a coordinate value of the display pointer to be beyond the bound of a bounded dimension.

At any time, an array element is either empty or it contains an element selected from one of the *character-repertoires* available within the current VTE. Such elements are called *character-box graphic elements* in the language of the VT base standards, but for simplicity they will simply be referred to as *characters* in this article.

continued on next page

The OSI Virtual Terminal ASE *(continued)*

There is no limit on the number of different character-repertoires that can be available for simultaneous use; this gives VT considerably more flexibility than the widely-used code extension facilities of ISO 6429 that specifies control functions for 7-bit and 8-bit coded character sets.

The content of the DO may be modified by a number of specific operations. The display pointer may be set by either an absolute or a relative addressing operation. A TEXT operation enters a character at the location addressed by the display pointer and then increments the x-coordinate of the display pointer by 1. There is no automatic line-feed; an addressing operation is required when TEXT operations increment the display pointer beyond the end of a bounded dimension. An ERASE operation is also available to cancel the character content of any array element so that it reverts to an empty state.

Attributes

In addition to its character content, each array element has a number of *rendition attributes,* namely font, emphasis, foreground-colour and background-colour. Values are necessarily assigned to each rendition attribute for any non-empty array element. When a TEXT operation enters a character into an array element, the rendition attributes are set according to the values of the *modal* and *global* attributes. These are two stand-alone sets of attribute values that may be set by ATTRIBUTE operations on the DO, but which may also take the symbolic value "null." The distinction between the two sets is that a non-null modal attribute value over-rides any value currently present in the array element concerned, but a non-null global attribute value is over-ridden by any value currently present. The global attribute value is thus of significance only when a character is entered into an empty array element. A default mechanism takes care of the case when both modal and global attribute values are null. The ATTRIBUTE operation may also be used directly to change the existing rendition attribute values of array elements.

Update windows

Real terminals vary greatly in their editing and cursor addressing capabilities. The simplest terminal may have no editing capabilities and may not permit any backward movement of the cursor. One step up from this is a terminal which enables the current line to be edited. Beyond this are terminals with full screen editing capabilities.

Such capabilities and restrictions are modelled in VT by a number of VTE parameters, which may be set independently for each dimension. All DO addressing operations may be allowed, or they may be restricted by the prohibition of absolute addressing operations or even of all explicit addressing operations. If explicit addressing operations are allowed, the display pointer may be restricted to forward motion only, or may be unconstrained.

In addition each dimension has an *update-window* which determines the region within which editing is permitted. Its size is specified by a VTE-parameter, and its upper edge is at the highest coordinate value that has been updated by a TEXT operation. A zero size prevents any editing of entered text. If the display pointer is moved to an address below the lower edge of the update-window, a TEXT operation will be invalid. If it is moved to an address above the upper edge of the update-window, a TEXT operation may be performed but will result in the update-window being moved forward.

Control objects

Operations on a display object can do nothing other than modify its content. An ASCII terminal, in contrast, reserves data values 0 to 31 for control purposes. If the value 7, representing the ASCII <BEL> character, is used in a VT TEXT operation it will simply be treated as a character and entered into the next array element.

Data exchanged for control purposes is modelled in VT by updates to *control objects* (COs) present in the VTE. There are many different types of control object, identified by their CO-type VTE-parameter. The base standards define several such CO-types, but they also permit additional CO-types to be defined within a VTE-profile. Additional CO-types which may be of use in a number of different profiles can be registered, to avoid repetition in each VTE-profile concerned.

Control objects can vary greatly in their complexity. The simplest COs have a single data element with a value from one of the categories "character," "boolean," "symbolic," "integer" and "transparent." The effect of the ASCII <BEL> control code may be achieved in VT by an update to an appropriate simple CO with the symbolic value "audible_alarm."

More complex control objects can have several data elements, which need not all be of the same category. Such COs are said to be *parametric,* since the data element to be updated may be identified by an integer parameter. There are also *non-parametric* COs, which are the most complex of all. There are no constraints on the data structure of a non-parametric CO. This structure is specified in the definition of the CO-type concerned. This definition must also contain the additions to the abstract syntax that are required in order to describe updates to this data structure.

Devices and device objects

The intention of VT is to isolate an application package from the details of the real terminal with which it is communicating. One VT-user will normally consist of a terminal system that has a real human user. Its display screen need not be large enough to image the entire display object at once. Provided that the image may be scrolled by local means through the current update-windows of the display object, the human user will be able to see all parts of the display object that he or she is permitted to modify.

The application package does not need to know the necessity for such scrolling. However, if it were aware of the real screen size, it may have the ability to modify its presentation in order to maximize the use of the available screen area. The real terminal may also not be able to distinguish all rendition attributes in its display, and so may need to resort to the use of codes or other means to indicate certain renditions. An application package that is aware of such limitations may be able to use alternative renditions to improve the display. Such information may be conveyed to the application package in the VTE-parameters of a *device object.*

Terminal updates to the display and control objects are considered as being performed by an *object updating device* of unspecified nature. It is the responsibility of the VT-user of the terminal end-system to translate actions at a real keyboard or other input device into the operations defined by the VT Service.

Real signalling devices such as an audible alarm may be associated directly with a control object. The CO update by the symbolic value "audible_alarm" considered in an above example may in this way operate a real alarm at the terminal end-system, as intended by the application package that generated the update.

continued on next page

The OSI Virtual Terminal ASE *(continued)*

Fields

The real power of VT comes from linking together the concepts of control object and display object. These have been described above as if they were two parallel but separate aspects of VT. In a simple VTE-profile this may indeed be the case. But by defining COs with semantics that augment the structure of the display object, an almost unlimited functionality may be provided. It is in such developments that the non-parametric COs become essential. The base standards use such COs to define a construct of *fields* for a display object.

The two-dimensional subset of the DO specified by a particular value of the z-coordinate is known as a *y-array.* A *field* is a subset of a single y-array whose array elements are addressed by a single coordinate, the *k-coordinate,* that takes values from 1 upwards. Several fields may be defined on a single y-array, but no array element may lie in more than one field. The fields of each y-array are themselves labelled by an *f-coordinate.* An array element within a field may thus be addressed uniquely by the triple (k,f,z) of coordinates as well as by its (x,y,z) coordinates. To distinguish these two addressing mechanisms, the triple (k,f,z) is known as the *logical address* and (x,y,z) as the *primitive address* of the array element. There are no constraints either on the subset of array elements that may constitute a field, or on the order in which they are addressed by the k-coordinate. Field definitions are stored in *Field Definition Records* (FDRs) that form the content of a non-parametric CO known as a *Field Definition Control Object* (FDCO).

The significance of fields is that the display pointer and all the associated addressing and DO update operations have parallel versions that use logical addressing. There is a *logical pointer* that contains the logical address of the array element that will be updated by a LOGICAL-TEXT operation. This operation will then increment the k-coordinate of the logical pointer. There are also explicit logical addressing operations as well as LOGICAL-ATTRIBUTE and LOGICAL-ERASE operations. The logical pointer and the display pointer are maintained independently; a LOGICAL-TEXT operation has no effect on the display pointer just as a TEXT operation has no effect on the logical pointer. Since it is the k-coordinate and not the x-coordinate that is incremented by a LOGICAL-TEXT operation, all the array elements of the field will be filled in order by successive LOGICAL-TEXT operations, regardless of the shape or arrangement of the field.

Navigation paths

A Field Definition Record is a composite value made up of several FDR-components. The specification of the array elements of a field forms only one FDR-component, known as its *field-extent.* Another is its *field-status,* which may take one of the three symbolic values "active," "inactive" and "non-extant." A field with "non-extant" field-status has no elements, all other FDR-components are "void," and no explicit record need be maintained. A field is deleted by setting its field-status to "non-extant." If instead the field status is set to "inactive" then the values of its other FDR-components are retained. It is in effect temporarily deleted, and may be restored simply by returning the field-status value to "active."

The *next-field* and *previous-field* FDR-components allow several fields to be linked together to form a *navigation path.* Each of these FDR-components is either "void" or it holds an f-coordinate value. These values are referenced by two special logical addressing operations, NEXT FIELD and PREVIOUS FIELD.

If the logical pointer lies in a field for which the next-field value is "void," then the NEXT FIELD operation moves the logical pointer to the first array element of the field with the next higher f-coordinate value. But if the next-field value is not "void," its value is instead used as the new f-coordinate value. The PREVIOUS FIELD operation works in an analogous manner.

A navigation path may be terminated by setting the next-field and previous-field values for the desired final fields to zero, or to the f-coordinate of a non-extant field. Re-entrant navigation paths are permitted, as are navigation paths in which NEXT FIELD followed by PREVIOUS FIELD does not return to the original field. Inactive fields are skipped by navigation paths, in that the NEXT FIELD and PREVIOUS FIELD operations are implicitly repeated until an active or non-extant field is encountered.

Field Entry Instructions

The most powerful feature of the field concept is provided by the *entry-control-list* FDR-component. This makes use of two further types of non-parametric control object associated with fields, namely *Field Entry Instruction Control Objects* (FEICOs) and *Field Entry Pilot Control Objects* (FEPCOs). The content of each of these COs is an indexed set of records known respectively as FEIRs and FEPRs. The entry-control-list for a field enables any number of FEIRs and FEPRs to be linked to that field by inclusion of their respective identifiers in the list.

An FEIR consists of a set of one or more *Field Entry Instructions* (FEIs). The particular FEIs that are permitted are specified as part of the definition of the FEICO concerned. Often an FEI will specify constraints on the data that may be entered into a field, such as by providing a list of allowed or disallowed characters or character strings, or by requiring that every array element of the field must be filled. FEIs may be parametric, so that the precise list of allowed or disallowed items may be specified separately for each field. Other possible uses of FEIs are:

- to specify the way that the location of the logical pointer should be indicated on a display device, which in more familiar terminology is a specification of the type of cursor to be used;
- to provide secret entry for passwords by requiring that characters entered into a particular field are not to be echoed to the display device;
- to implement a menu system that enables signalling not by the entry of data into a field, but instead by the selection of one field from a number that have been designated as "selectable" by linkage to an appropriate FEI.

Field Entry Pilots

The ability to specify constraints on data entry into a field implies the possibility that these constraints may be violated. Such a violation is one example of a *Field Entry Event* (FEE), which as its name implies is simply an identifiable event associated with data entry into a field. The selection of a "selectable" field is another possible FEE, as is the entry of a character into the last array element of a field. Such FEEs are processed in accordance with the FEPRs linked to the field concerned. An FEPR comprises:

- a Field Entry Event (FEE);
- a set of Field Entry Conditions (FECs);
- a sequence of Field Entry Reactions (FERs).

continued on next page

The OSI Virtual Terminal ASE *(continued)*

When an FEE occurs during data entry into a particular field, and there is a FEPR linked to that field which references the FEE, then the conditions expressed by its FECs are tested. If all the conditions are satisfied then the sequence of reactions specified by the FERs will be performed. As for the FEIs of a FEICO, the FEEs, FECs and FERs that are permitted are specified as part of the definition of the FEPCO concerned and all of these items may be parametric.

When a terminal enters a character into the last array element of a field, a common requirement is to move to the first array element of the next field in the navigation path. If the current field is already the last field of the navigation path, then instead control is to be returned to the application package. The use of FEPRs enables such behaviour to be programmed into the terminal by the application package.

This example shows that it is often useful to have alternative reactions according as a particular condition is or is not satisfied. The basic construction of a FEPR suggests that this requires two FECs to be defined, each being true when the other is false, and two FEPRs to be linked to the field, each testing one of the two conditions. However, the ability to define parametric FERs enables FERs to be defined that take FECs and other FERs as parameters. Each parameter may be given an identifier, so that it is possible to define an FER that takes three parameters with identifiers "if," "then" and "else." The "if" parameter takes an FEC as value while the "then" and "else" parameters each take a sequence of FERs. The semantics of this FER is that if the "if" FEC is satisfied then the "then" FERs are performed, otherwise the "else" FERs are performed. The specification of the FER then reads naturally in terms of the assigned identifiers, and the above example can be achieved with a single FEPR and with the need to define only a single FEC.

Even greater flexibility can be achieved by the use of recursive FERs. It is possible for an FER to initiate an FEE and thus to cause further FEPR processing. This ability can be combined with the "if then else" construction to define FEPRs that repeat some action until (or while) a specified condition is satisfied. A simple example would be to implement a field search operation by incrementing the k-coordinate until the logical pointer reaches a specified character. The FEPCO defined within the VT Forms Profile provides these facilities and is largely responsible for the considerable power and versatility of this Profile specification.

Modes and access rules

Although the concept of a display object is modelled on the screen of a real terminal, within VT it is not considered as located within either VT-user. It resides instead in a shared *Conceptual Communication Area* (CCA) to which both VT-users have access, and which in addition contains the control and device objects and all the data structure definitions that form part of the current VTE.

As the liberal use of the terms "virtual" and "conceptual" imply, neither the CCA nor its contents have any real existence. In particular the communication facilities are not required to provide storage to realise the display and control object contents. Communication instead takes place by one VT-user issuing updates to the CCA and these updates then being passed to the other VT-user by means of the VT Protocol. Both VT-users are thus provided with the information required to maintain their own realizations of the CCA if they so require.

Such a procedure requires access controls to ensure that both VT-users do not try to update the same object at the same time. There are two modes of operation provided by VT to achieve this. In *S-mode* (synchronous mode) operation there is a single display object together with a transferrable access-right known as WAVAR (*Write Access VARiable*). In *A-mode* (asynchronous mode) operation there are two display objects, one with WACI (*Write Access Connection Initiator*) and the other with WACA (*Write Access Connection Acceptor*) access-right, both access-rights being non-transferrable. In S-mode, facilities are available for the VT-user currently owning WAVAR to relinquish it, and for the VT-user not currently owning WAVAR to request it.

Each control object also has an access-right, the value being specified as part of its definition. There is a greater range of possible access-rights for control objects, for in addition to WAVAR, WACI and WACA there is also NSAC (*Not Subject to Access Control*), the composite values WAVAR & WACI and WAVAR & WACA, and even the possibility of no-access. The values involving WAVAR are only available in S-mode, WAVAR & WACI meaning for example that the connection acceptor never has access and that the connection initiator has access only when also owning WAVAR. The value of "no-access" is appropriate, for example, for a FEICO or FEPCO whose content is fully specified within the VTE-profile definition.

Terminal and Application asymmetry

The use of appropriate access-rights is particularly important for the control objects associated with the use of fields. In S-mode field usage the application package will wish to control the field definitions and will normally also wish to prevent the terminal from updating array elements that lie outside any defined field. At the same time the application will wish to retain its right to update any part of the display object.

To achieve these aims the FDCO is given one of the access-rights WAVAR & WACI or WAVAR & WACA. The VT-user which is able to update the FDCO when it owns WAVAR is then designated the *Application VT-user,* the other being designated the *Terminal VT-user.* This is the only circumstance in which the VT base standards recognise an asymmetry between the VT-users, although of course a *de facto* asymmetry may exist in other circumstances. The FEICOs and FEPCOs either have the same access-right as the FDCO or else have no-access.

In the asymmetric case the Terminal VT-user can use logical operations only on active fields, but the Application VT-user may also use them on inactive fields. In addition a VTE-parameter known as "access-outside-fields" may be set to one of the values "allowed" and "not allowed." If it takes the value "not allowed," then the Terminal VT-user is restricted to logical operations on the display object while the Application VT-user is permitted to use both logical and primitive operations.

The Profiles in outline

This final section gives a summary of the internationally agreed VT Profiles that are currently defined or are under active development. They are listed together with their Profile Identifiers as specified in the taxonomy of ISPs given in ISO/IEC/TR 10000-2 [7].

AVT 12, A-mode TELNET: This Profile provides capabilities equivalent to those of a real TELNET terminal. It includes a model of the TELNET "Network Virtual Terminal" using VT concepts. It is primarily intended to ease the migration of TELNET environments to full OSI working. Fields are not used.

continued on next page

The OSI Virtual Terminal ASE *(continued)*

AVT 13, A-mode Line Scroll: This Profile is still under development. It is designed to support line-at-a-time interactions between terminal and application end-systems, as typified by operating system command entry. Line scrolling is bi-directional, earlier lines may be edited within the update-window and "type-ahead" is supported. Fields are not used.

AVT 15, A-mode CCITT X.3 PAD Interworking: This Profile provides capabilities equivalent to those of a character terminal connected to a CCITT X.3 PAD. It includes a model of the X.3 PAD parameters and associated X.29 operations using VT concepts. It is primarily intended to ease the migration of X.29 environments to full OSI working. Fields are not used.

AVT 16, A-mode Transparent: This provides a transparent mode of operation which allows VT-users to exchange transparent uninterpreted binary character streams. It is primarily intended for use when VT-users wish to control real terminals directly through the use of embedded control characters. Fields are not used.

AVT 22, S-mode Forms: This is intended for use in a full OSI environment. It makes full use of the capabilities of the field concept and includes a powerful FEICO and FEPCO that are fully defined within the VTE-profile. It provides a general-purpose full screen terminal capability with particular provision for the facilities required in full screen editing and in the entry of data into forms.

AVT 23, S-mode Paged: This Profile is still under development. It is expected to comprise those facilities of the S-mode Forms Profile that are practicable when the real terminal attached to the Terminal VT-user operates on a block or paged basis rather than on a character by character basis.

References

[1] Marshall T. Rose, "Components of OSI: The Application Layer Structure," *ConneXions,* Volume 4, No. 1, January 1990.

[2] "Information technology—Open Systems Interconnection—Virtual Terminal Basic Class Service," International Standard ISO 9040, 1990.

[3] "Information technology—Open Systems Interconnection—Virtual Terminal Basic Class Protocol—Part 1: Specification," International Standard ISO 9041-1, 1990.

[4] David Chappell, "Components of OSI: The Presentation Layer," *ConneXions,* Volume 3, No. 11, November 1989.

[5] "Stable Implementation Agreements for Open Systems Interconnection Protocols, Version 3, Edition 1," NIST Special Publication 500-177, December 1989.

[6] David Chappell, "Components of OSI: A taxonomy of the players," *ConneXions,* Volume 3, No. 12, December 1989.

[7] "Information technology—Framework and taxonomy of International Standardized Profiles—Part 2: Taxonomy," Technical Report ISO/IEC/TR 10000-2, 1990.

GRAHAM DIXON is Director of Studies in Mathematics at Churchill College, Cambridge, England. He also acts on behalf of the College as a consultant on OSI. He is a member of the NIST OIW Special Interest Group on VT and of the EWOS Expert Group on VT. He is editor of the EWOS VT S-mode Forms Functional Standard and of the ISP AVT 22 that is being developed from it. He received his Ph.D. degree in mathematics in 1965 and is the author of a book on Special Relativity published by Cambridge University Press.

EUUG Becomes *EurOpen*

Almost 15 years after its creation, *EUUG* (European UNIX Users Group) has changed its name to *EurOpen,* the European Forum for Open Systems, to reflect the progressive expansion of the group to encompass Open Systems.

Background

EUUG was created in 1976 by a group of European UNIX systems programmers. It has since grown to become a major European association of National Groups, with representatives in all European countries. The change of name marks another significant step in the group's history which parallels the evolution that UNIX has taken over the past few years, leading to the concept of Open Systems.

"This change of name, reflects the development of the European UNIX community which has come to maturity over the past few years. It testifies to the significant contribution that UNIX has brought to the computer industry in general, and at the same time opens the way to the era of Open Systems far beyond the early vision of the UNIX Pioneers," said Michel Gien, the EurOpen Chairman.

EurOpen has extended the relationships between itself and other organizations related to UNIX and Open Systems outside Europe such as the *USENIX Association* and *UniForum.* EurOpen and UniForum are jointly organizing *OpenForum,* the largest European conference and exhibition in the Autumn of 1992 in Amsterdam, the Netherlands.

Members

EurOpen (as was the EUUG before it) is a federation of National UNIX and Open Systems Users Groups from Europe and beyond. Today there are 20 National Groups and the number is rapidly growing. EurOpen operates via a two level organizational structure: a Governing Board with a representative of each National Group deciding on general directions, and an Executive Committee in charge of actual operations. EurOpen members include Austria, Belgium, Czechoslovakia, Denmark, Finland, France, Germany, Hungary, Iceland, Ireland, Italy, The Netherlands, Norway, Portugal, The Soviet Union, Spain, Sweden, Switzerland, The United Kingdom, and Yugoslavia.

Expansion

EurOpen continues to serve the UNIX and Open Systems community, building upon its current services including publications, conferences, software distributions, reports on Open Systems standards, the *EUnet* and *InterEUnet* networks, and more. EurOpen will expand the quality and scope of those services to cover all aspects of Open Systems, including technical as well as strategic and market interests. This expansion includes the coordination between the EurOpen National Groups activities and the administration of EUnet and InterEUnet, the European UNIX networks.

Goals

EurOpen, the European Forum for Open Systems is an international federation of UNIX and Open Systems Users groups. EurOpen is dedicated to (1) promoting and advancing the knowledge, use, and application of open computer systems using compatibility techniques pioneered by the UNIX operating system, (2) facilitating the exchange of information and views on the use and development of open systems, (3) informing the public on open systems, (4) providing a focus for the standardization of techniques used in open systems, and (5) encouraging internationalization of open systems, benefiting users of open systems.

Congestion Avoidance

by Paul E. McKenney, SRI International *

Introduction

This article presents a survey of congestion avoidance in computer communications networks, showing how congestion avoidance relates to other fields and giving a brief overview of work in the area.

The purpose of congestion avoidance is to promote efficient and fair use of network resources, in spite of overload conditions and abuse from users who might be uncooperative or just plain malicious.

Unfortunately, it is just as difficult to define efficiency, fairness, and overload in networking as it is in any other human endeavor. The following rough-and-ready definitions have nevertheless proven quite useful:

- *Efficiency:* The most efficient protocols get the most useful work done using a given set of resources. We will arbitrarily assume that all user data packets *except* retransmissions accomplish useful work. If you find this assumption very difficult to swallow, consider that one man's trash is another man's treasure.

- *Fairness:* Efficiency and fairness often conflict, as shown in Figure 1. Here users 2, 3, and 4 do not compete among themselves for resources (the links connecting nodes A, B, C, and D), but user 1 competes with each of users 2, 3, and 4. Increasing user 1's throughput by one unit forces each of users 2, 3, and 4 to decrease their throughput by one unit, for a net loss of two units of network throughput. Thus, the most efficient resource allocation favors users 2, 3, and 4 to the total exclusion of user 1. This results in 50% more throughput than if each user were treated equally.

  We will use the "max-min" definition of fairness [1], so named because it gives the maximum possible level of service to the users who will receive the minimum.

  A protocol that enforced max-min fairness would be a perfect networking Robin Hood—it would rob from the rich and give to the poor, taking care that the victims fare no worse off than the beneficiaries and that the beneficiaries can make good use of their windfalls.

- *Overload:* Each application has its own definition of overload. For example, a network that appears grossly overloaded to a user of an interactive application might seem just fine to an electronic mail user. We will nevertheless define overload to occur when the load offered to the network exceeds its capacity. Some recent algorithms are able to work with more sophisticated definitions; these algorithms are discussed near the end of this article.

Relation to other fields

Congestion avoidance has strong ties to many other fields. The rest of this section gives an overview of the relation between congestion avoidance and control theory, transportation, and telephony.

Control theory provides conceptual tools that engineers have used to tame electrical, mechanical, hydraulic, and pneumatic systems over the past century. It is natural to presume that these tools could be easily used to tame computer networks. Unfortunately, networks pose some special challenges.

** Ed.: Mr. McKenney is now with Sequent Computer Systems.*

First, networks do not always obey a nice set of physical laws; packets can be created, duplicated, and destroyed arbitrarily on many types of networks. (Life would certainly be less pleasant if water (to say nothing of less agreeable substances) had a similar disdain for the law of conservation of matter).

Second, networks consist of an extremely large number of very complex active components. Although very small or very regular networks with simple protocols and traffic patterns have been successfully analyzed, it seems unlikely that anyone will succeed in carrying out an exact analysis of the Internet in the near future.

Finally, the very large delay-bandwidth products inherent in high-speed wide-area networks pose difficulties for simple control theory approaches. For example, imagine a 1-Gbps link running from Boston to San Francisco. In the absence of some advances in theoretical physics, the link's round-trip delay will be at least 30 milliseconds. This means that the node in San Francisco will have the opportunity to transmit almost 4 megabytes of data before it can have any chance of learning whether the first bit of data arrived successfully in Boston. And if you think that this is a problem, just wait until we have Internet sites on the Moon (with a round-trip time of over two seconds) or Mars (with a round-trip time varying from not quite 9 to more than 42 *minutes*). Despite these difficulties, researchers have made great strides toward the goal of taming computer networks with control theory tools [6].

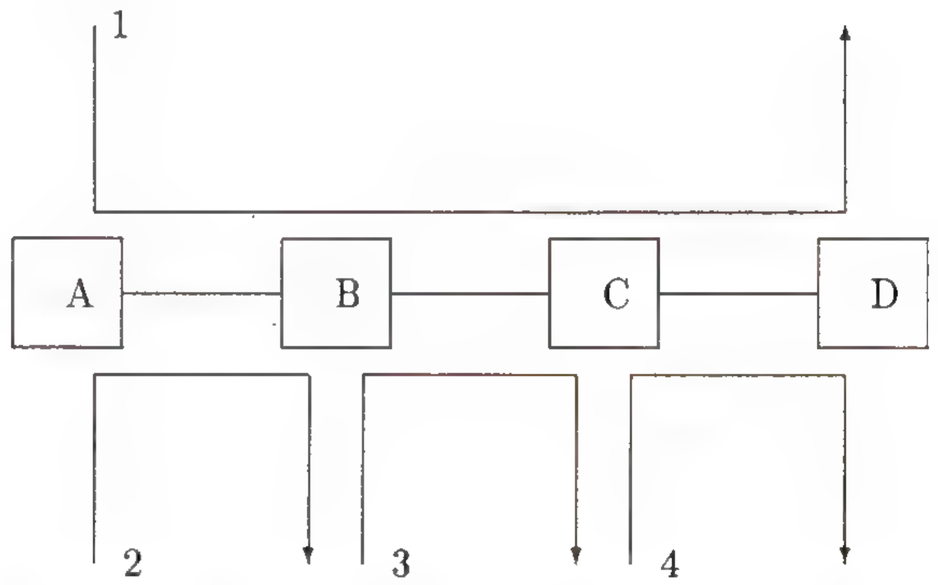

Figure 1: Conflict between efficiency and fairness

Analogies

The analogy between vehicular traffic and computer network traffic is seductive: cars, trains, or planes take the place of packets; roads, rails, or airspace segments take the places of links; and traffic intersections take the place of gateways. However, there are some important differences.

For example, it is possible to communicate the state of a vehicular transportation system to a central location very quickly compared to the rate of change in the state of the system. For example, a traffic reporter can describe conditions quickly and effectively to individual drivers over their car radios. If commuter traffic were subject to the same restrictions that many networks are, drivers could find out about traffic conditions only by asking other drivers who had recently been in the area of interest.

On the other hand, when a computer network becomes too congested, the excess packets can always be discarded. In contrast, many people believe that grinding up excess cars in order to relieve traffic congestion is socially irresponsible.

continued on next page

Congestion Avoidance *(continued)*

There is a large body of experience in control of telephone networks. Unfortunately, applying this experience to computer networks is difficult for two reasons. First, each voice telephone user places a negligible load on the phone system. In contrast, a single undergraduate student can easily absorb the entire bandwidth of an Internet link, especially if there is no competing traffic. Second, while a single phone conversation places a highly predictable load on the system, computer network users tend to present comparatively bursty loads. Although in the past this has prevented telephony experience from being applied to computer networks, the situation is likely to change dramatically as telephone companies move to support high-speed data communications. In summary, although much insight can be gained from other fields, congestion avoidance presents a unique set of challenges.

SMDS

Telephone companies are moving to support services such as *Switched Multimegabit Data Service* (SMDS). To help ensure success, they will need to bring their substantial research resources to bear on computer communications problems. This is likely to lead to exciting advances in the field over the next few years.

Early theoretical work

Researchers did substantial work in the area of congestion avoidance in the early 1980s. (See Bertsekas [1] for an overview and bibliography). Much of this work attacked the problem from an analytic viewpoint, and culminated in some algorithms that converge to fair resource allocations and that never result in overloaded links. Unfortunately, these algorithms require global knowledge of the traffic load offered by each user, which is usually not available. In addition, the iterative optimization techniques typically employed by these algorithms require excessive computational resources, even on today's high-performance microprocessors.

Currently deployed algorithms

The congestion-avoidance algorithms in common use today are much simpler. Although they do not necessarily converge to perfectly fair resource allocations, they do a much better job of coping with the dynamic traffic conditions found in production networks than would the earlier algorithms.

Slow-start

Van Jacobson's *slow-start* TCP is by far the most widely used congestion-avoidance algorithm. The following is a very cursory overview of this important algorithm; readers interested in an in-depth treatment should refer to Jacobson's SIGCOMM '88 paper [6]. The algorithm relies on several properties of current networks: (1) window transport protocols such as TCP have a self-clocking property, (2) bit-error rates are relatively low, so packet loss is due mostly to congestion, and (3) as traffic load rises, per-packet delay becomes greater and more erratic.

TCP uses acknowledgments and retransmissions to attain reliable transfer of data. Waiting for an acknowledgement after transmitting each and every packet would often result in wasted network bandwidth due to the long round-trip times on wide-area networks, so TCP will transmit several packets (the exact number is controlled by the current window size) before waiting for acknowledgements. Under normal conditions, acknowledgements are sent immediately upon reception of packets, and new packets are sent immediately upon reception of acknowledgements. The slowest link will control the rate at which acknowledgements arrive at the transmitting node; this is the basis for the self-clocking property.

Adjusting the window size

Another crucial factor is the size of the window. TCP will exhibit oscillating behavior if the window is too small—first TCP will send as many packets as the window allows, properly spaced for the slowest link, then TCP will sit idle waiting for the first acknowledgement to arrive, after which it will send another group of packets to repeat the cycle. If the window size is "just right," the first acknowledgment will arrive shortly after the window is exhausted. The result will be a nice, steady, evenly spaced stream of packets, flowing exactly at the rate allowed by the slowest link. Increasing the window size beyond this point will not increase the packet flow rate, since the slowest link is already going as fast as it can. Rather, the larger window size results in excess packets piling up in the queue feeding into the slowest link. The more the window size is increased, the larger that queue becomes, until the queue overflows.

If the queue overflow results in the loss of just one packet and if the window is sufficiently large, the arrival of each subsequent packet at the receiving node will generate an acknowledgement of only those packets preceding the lost packet. The arrival of these identical acknowledgement packets will cause TCP to retransmit the lost packet, and then to continue on, with only a small gap in the packet stream.

However, if too many packets are lost, or if the window is too small, TCP will stall waiting for the acknowledgement for the lost packets. Of course, TCP does not wait forever; after a period of time called the "retransmission timeout" (measured from the time at which the lost packet was initially transmitted), the packet that was lost is retransmitted. If the window size were to remain fixed, the result would be bursts of packets resulting in queue overflow, separated by pauses due to the retransmission timeout. This is a symptom of congestion. A large part of Jacobson's algorithm consists of adjusting the window size to avoid this undesirable behavior. (Jacobson's TCP actually uses two windows, one to guard against congestion, the other to guard against buffer overflow at the destination. In this article, we will assume that the destination has enough memory so that buffer overflow never occurs, and consider only the window that guards against congestion).

Since today's networks typically have very low loss rates due to bit-errors, it is safe to assume that if a packet was lost, it was lost due to queue overflow, in other words, due to congestion. This means that TCP can take a lost packet to be a hint that the window was too large and should be made smaller—reduced by half, to be precise. In theory, one might try to analyze the time intervals between the arrival of acknowledgements in order to determine whether to window should be made larger. In practice this would require excessive computation and could be easily fooled by variations in the amount of data that the user had available to send. A better approach, used by Jacobson's TCP, is to periodically increase the window size by one packet if no packets are lost.

Retransmission timeout

The value of the *retransmission timeout* is just as crucial to smooth and efficient operation as the window size is. Too large a value for the retransmission timeout will result in excessive time spent waiting for lost packets—and since the window size is inexorably increased until packets are lost, packet loss is inevitable. Too small a value will result in unnecessary retransmission of packets.

continued on next page

Congestion Avoidance *(continued)*

Since the round-trip time is subject to variations due to statistical fluctuations in traffic load, and since these variations grow larger as the load increases, the value chosen for the timeout must reflect both the average value and variance of round-trip time.

To see this, imagine that you are waiting for two friends, one slow and punctual, and the other quick but easily distracted. Suppose that you expect your slow and fast friends to take ten and five minutes, respectively, to arrive. How long must you wait before giving up on them? After fifteen minutes, you might be very sure that your slow friend has stood you up. You might have to wait an hour or more before you could be so certain that your fast friend had abandoned you. With networks, as with friends, average speed does not tell the whole story.

In summary, Jacobson's TCP exploits the self-clocking property to obtain smooth data flow. It relies on the fact that bit-error rates are very low in order to use packet loss as a reliable indication of congestion. Finally, it sets retransmission timeouts based on both the average value and the variance of round-trip time.

This description has only touched the surface of this protocol. In particular, it has not covered the "slow-start" phase of the protocol, or the method for estimating the mean and variance of the round-trip time. To learn more, refer to Jacobson's SIGCOMM '88 paper [6] and RFC 793 [12]. Of course, for the truly adventurous, there is always the source code to UNIX 4.3BSD-reno.

This algorithm is widely used and has done much to promote efficient use of Internet resources. Nonetheless, this algorithm is not without shortcomings. It does not guarantee fair resource allocation between competing users, nor does it work well over networks with high bit-error rates or high bandwidth-delay products. In addition, it has a very narrow definition of network overload—the network is assumed to be overloaded when it begins to drop packets. It seems safe to assume that Van Jacobson is working to overcome these shortcomings.

DECbit

Another protocol that has seen significant use in production networks is the DECbit algorithm [7] used in DECnet and OSI CLNS. This algorithm uses a congestion window in a manner similar to Jacobson's TCP. One major difference is that DECbit makes use of a bit in each packet that is set by any congested gateway that the packet passes through. If more than half of the packets that arrive at the destination within a period of two round-trip times have their bit set, the window size is reduced to 87.5% of its previous value, otherwise the window size is increased by one packet. Most of the experience with this protocol has been in low-speed networks; work is in progress to evaluate its performance in modern high-speed networks. An attractive aspect of this algorithm is the possibility of controlling congestion without relying on packet loss; it will be interesting to see if this possibility is realized in high-speed networks.

BBN Algorithm

Finally, the BBN algorithm [13] is an example of a rate-based congestion-avoidance algorithm. Each gateway maintains a rate limit that applies to any stream of packets passing through it; the gateway periodically multiplies this limit value by the ratio of the target utilization to the actual utilization (e.g., a link's target utilization might be 80% of its capacity).

This has the effect of decreasing the limit when resources are over-utilized, and increasing the limit when resources are underutilized. The rate limit is communicated to the source nodes by piggybacking it on data and acknowledgement packets in the same manner as DECbit piggybacks the congestion-experienced bits. In the absence of multi-path routing algorithms and bursty traffic, this algorithm will converge to the max-min fair resource allocation. This algorithm has seen use in low-speed, homogenous networks such as the Defense Data Network (DDN) and in BBN internal networks; we may see how well it handles higher-speed networks if the DDN moves to T1-speed links.

Future prospects

Current work in congestion avoidance is in two main areas, improving the fairness of currently deployed algorithms [8], and accommodating applications such as real-time voice and video that have stringent throughput and delay requirements.

Researchers have proposed the use of *fair queuing* [3, 11] to improve the fairness of currently deployed algorithms. Concerns about the computational demands of strict fair queuing have lead to development of a probabilistic algorithm, stochastic fairness queuing, that closely approximates fair queuing's behavior [10]. Simulations of both fair queuing and stochastic fairness queuing have shown them to be very effective in achieving fair resource allocations in a variety of topologies and traffic patterns.

Applications such as real-time voice and video have stringent, but predictable throughput and delay requirements. This has lead some researchers to propose that such applications reserve network resources [2, 4, 5, 9, 15]. Such reservations would allow the network to: (1) estimate future load, (2) refuse traffic that it could not carry, and (3) make performance guarantees to the application. This approach allows each application to define the quality of service that it requires, in effect to make its own definition of network overload. There is much work to be done in this area.

FEC

The stringent delay requirements of real-time voice and video have also led some researchers to propose use of *Forward Error Correction* (FEC) to reduce the need for the retransmissions that would otherwise be required to recover packets lost due to congestion. The idea is to send a small number of additional packets that provide sufficient redundant information to enable the receiver to reconstitute lost packets. However, these additional packets increase the load on the network, which in turn increases the packet loss rate. Thus, the FEC scheme must not only be capable of reconstituting a significant number of the lost packets, it must also be capable of reconstituting the additional packets lost because of the transmission of the packets containing the redundant information.

Preliminary results indicate that the use of simple FEC codes, combined with selective rejection of packets from node buffers, can result in a reduction of up to three orders of magnitude in the packet loss rate [14].

The new possibilities being opened up by the prospect of gigabit networks and the increasing level of interest in the field will without doubt cause the field of congestion avoidance to continue to be quite exciting and productive.

continued on next page

Congestion Avoidance *(continued)*

References

[1] D. Bertsekas & R. Gallager, *Data Networks,* Prentice-Hall, 1987.

[2] Douglas E. Comer & Rajendra Yavatkar, "Flows: Performance guarantees in best effort delivery systems," Technical Report CSD-TR-791, Purdue University, July 1988.

[3] Alan Demers, Srinivasan Keshav, & Scott Shenker, "Analysis and simulation of a fair queuing algorithm," Proceedings of *SIGCOMM '89,* 1989.

[4] Domenico Ferrari, "Real-time communication in packet-switching wide-area networks," Technical Report TR-89-022, International Computer Science Institute, Berkeley, CA, 1989.

[5] COIP Working Group, "ST-II specification," Working document for IETF COIP Working Group, April 1990.

[6] Van Jacobson, "Congestion avoidance and control," Proceedings of *SIGCOMM '88,* pages 314–329, August 1988.

[7] Raj Jain & K. K. Ramakrishnan, "Congestion avoidance in computer networks with a connectionless network layer," Technical Report DEC-TR-506, Digital Equipment Corporation, 1987.

[8] A. Mankin & K. K. Ramakrishnan, "Gateway congestion control survey," Working document for IETF Congestion Control Working Group, June 1990.

[9] Tony Y. Mazraani & Gurudatta M. Parulkar, "Specification of a multipoint congram-oriented high performance internet protocol," Technical Report WUCS-89-20, Washington University, Saint Louis, Missouri, September 1989.

[10] Paul E. McKenney, "Stochastic fair queuing," Proceedings IEEE *INFOCOM '90,* San Francisco, June 1990.

[11] John Nagle, "On packet switches with infinite storage," IEEE *Transactions on Communications,* March 1987.

[12] Jonathan B. Postel, "Transmission Control Protocol," RFC 793.

[13] John Robinson, Dan Friedman, & Martha Steenstrup, "Congestion control in BBN packet-switched networks," *Computer Communications Review,* Vol. 20, No. 1, January 1990.

[14] Nachum Shacham & Paul E. McKenny, "Packet Recovery in High-Speed Networks using Coding and Buffer Management," Proceedings IEEE *INFOCOM '90,* San Francisco, June 1990.

[15] Lixia Zhang, "A New Architecture for Packet Switching Network Protocols," PhD thesis, Massachusetts Institute of Technology, 1989.

PAUL E. McKENNEY received B.S. (1981) degrees in mechanical engineering and in computer science and the M.S. (1988) degree in computer science from Oregon State University. From 1981 through 1985 he worked as a self-employed contract programmer and consultant, primarily in the area of real-time control systems. He joined SRI International in 1986 as a systems programmer. In 1987, he joined the Information and Telecommunications Sciences and Technology Division, where he was involved in measurement and control of dynamic computer communications networks. In September of 1990 he joined Sequent Computer Systems in Beaverton, Oregon as a Senior Engineer. Mr. McKenney is a member of the Internet End-To-End Research Group and the Internet Engineering Task Force.

Letters to the Editor

Ole,

Over the last few years, the issue of how to do distributed mail support within a site full of single-user workstations and PCs has attracted a good deal of attention from protocol designers, but little from commercial vendors. Protocol systems addressing this include POP2/SMTP, PCMAIL (RFC 1056), POP3/SMTP and POP3/IMAP. Functionality and details differ, but each combination manages to give the user mail access without requiring either a single central machine big enough for everyone to log in on, or the installation and maintenance nightmare of making each host act as a full mail forwarder. (POP stands for the *Post Office Protocol*).

Many clients

At the moment, there are probably almost a dozen supported commercial clients for POP2, POP3 and PCMAIL on various platforms including DOS, OS/2 and MACs. However, I did a survey last summer and only turned up one new supported server (POP2) to add to the two commercial POP2 servers I already knew of. This seems somewhat lame, given the availability of mature freeware servers for all the protocols, and the great utility distributed mail represents (I'm on a commuter train using a laptop to edit this letter. It will get sent once I get to work. This is how I handle almost all of my mail).

Need servers

Why am I saying this? Not as a commercial, or a boast; Sun and IBM offered DOS clients to go with their servers long before we did, the first three OS/2 TCP/IPs to ship all included clients, several Macintosh clients exist. Instead, I think distributed mail is an idea whose time has come, and I would be pleased to see it as widespread as The X Window System in a couple of years. This won't happen unless supported servers are widely available.

Push your vendor!

So, the message: Distributed mail is here, it works. Users and managers who want it supported should bend their vendors' ears. Vendors should look around and evaluate the market that's developing, the number of clients that are available. POP2 is obsolescent, I personally prefer PCMAIL, others prefer POP3 with IMAP, but in the long run timely action seems more important to me than which protocol system eventually wins. Read the RFCs, peruse the freeware and get off the dime... *—James B. VanBokkelen, FTP Software*

[Ed.: Speaking of electronic mail, we plan to pursue this topic further in future issues of ConneXions. Articles will include an overview of the X.400 Message Store, some experiences with X.400 and X.500 pilot projects, as well as a look at the Intermail project at USC-ISI. The interworking between electronic mail and FAX systems is also a topic which is receiving some attention in the networking community. We will keep you informed as developments happen].

continued on next page

Letters to the Editor *(continued)*

Dear Editor:

I hope you'll permit me to flame a bit on a subject near and dear to my pocketbook—the high cost of international standards documents.

With my new user-friendly accounting system, I'm able to finally provide a definitive answer to a question I've been somewhat curious about. My current outlay for OSI and CCITT documents is $3,734.78, which includes a 50% discount on many OSI documents because I flew to Geneva to pick them up.

High cost

I don't have the whole set, by any means. Getting something approaching a complete set will easily cost in the five figures. This may seem like a normal cost of doing business to some, but I'm afraid that the high cost of documents is having a real effect on their acceptance. That's bad if you happen to believe that the standards process can contribute something valuable.

OSI, ANSI, and CCITT all try to use document sales to subsidize their efforts. Remember, if they don't charge the general public for services, they will have to look someplace else for a source of funding for their activities. High costs to the general public for documentation is a way of defraying the cost to members.

In the US there is an interesting business arrangement for sales of standards documents. OSI has given ANSI an exclusive license to sell their documents in the US. ANSI, in turn, has delegated that privilege to groups like OMNICOM. Of course, ANSI sets the prices as part of their sublicensing deals, so don't expect a big price war in the near future.

The prices for documentation is not cheap. When you look at my $3,734.78 expenditure, keep in mind that this is a fairly small stack by computer standards. When I got one $1,354 pile from OMNICOM, it worked out to $388 per inch. My fairly complete set of the TCP/IP RFCs, by contrast, cost only a hundred dollars for connect time to UUNET and yielded two file cabinets full of documents.

The high cost is possible because ISO, ANSI, and CCITT retain the copyright on their standards. Being a standards body, they are, by definition, a monopoly supplier. By definition, if you want to know about the standards you have to pay the price.

This may seem like a trivial issue if you are used to having relevant standards delivered to your desk in the course of your work. Not everybody is so lucky.

Effects

Think about somebody trying to finish an B.S. or M.S. in computer science. I would think that a great topic these days would be something like "An Investigation of FTAM Performance on a Multiprocess System" or "Transmission of ODA Documents Across Multiple Messaging Environments," or a host of other OSI-related topics. Of course, these types of dissertations increase the pool of qualified programmers, fuel startups, and make an incredible wealth of public domain software available.

Students have no right to go to the library and make a copy of the relevant standards document. Even worse, the student has no place to send mail or to FTP the relevant document. "Copyright" means "no copying unless permission is granted," and it is *explicitly withheld* in this case.

Students are not the only population that should have easy access to documentation. Programmers at a company should be able to grab a document in their spare time and increase their skill levels. When I teach seminars, I'd love to include the standards documents as part of the hand-outs. This gives me the luxury of suggesting that students with questions "Read the FTAM Manual" (RTFM) as a more comprehensive source of information than any class or magazine article can ever provide.

How are the standards ever going to become generally accepted unless people can easily find out about them? There is absolutely no excuse for OSI, CCITT, and ANSI documents to not be available via FTP and mail servers in the same way we can get the RFC series.

No comment

The argument commonly given is that documentation sales help fund the standards process. I did a little checking on this point, and I found that both ISO and ANSI refused to comment on the finances involved. The percentage of the standards process funded by documentation sales, the percentage overhead allocation, and other such pieces of information were considered "proprietary."

Proprietary standards? That's kind of like "secret laws." How can something be a standard unless people have easy access to the information? Charging $10 per page for a document certainly doesn't count as easy access in my book!

Alternative funding

What seems to be needed is an alternative way of funding the standards process. An easy solution is to give more of the costs to the participants and less from documentation sales. Of course, people will argue that this makes the standards process harder to participate in by raising the cost to small companies.

How about charging big companies more? Getting the government more involved? Licensing standards if they get used? Reducing the bureaucracy at the standards agencies?

I'm not saying I have the answer to this problem, but it sure seems that there ought to be a more wide-ranging discussion and a realization that easy access to information should be at the heart of any standards activity.

—Carl Malamud

Ed.: We appreciate the comments from Mr. Malamud and invite anyone with "opposing views" to write us. Coincidentally, there is just enough room left on this page to remind you once again how to get RFC documents:

Getting RFCs

RFCs can be obtained via FTP from host `nic.ddn.mil` with the pathname `RFC:RFCnnnn.TXT` (where "nnnn" refers to the number of the RFC). FTP with, username `anonymous` and password `guest`. The NIC also provides an automatic mail service for those sites which cannot use FTP. Address the request to: `Service@nic.ddn.mil` and in the subject field of the message indicate the RFC number, as in "Subject: RFC 1175." Contact the NIC for information on obtaining hardcopy versions of RFCs. Their telephone numbers are: 1-800-235-3155 or 1-415-859-3695.

Requests for special distribution should be addressed to either the author of the RFC in question, or to `nic@nic.ddn.mil`. Unless specifically noted otherwise on the RFC itself, all RFCs are for unlimited distribution.

Submissions for Requests for Comments should be sent to: `Postel@isi.edu`.

Book Review

The Art of Computer Systems Performance Analysis; Techniques for Experimental Design, Measurement, Simulation and Modelling by Raj Jain, John Wiley and Sons, 1991, ISBN 0471-50336-3

The goal of this large book is to comprehensively survey the field of computer performance analysis.

Organization

The book starts with an brief overview (3 short chapters) of performance evaluation, which lists key journals, discusses common errors made in performance evaluation, and touches on issues of choosing evaluation techniques and metrics. The second section of the book (7 chapters) presents measurement techniques and tools. There are good discussions of issues related to choosing workload models characterizing results from workload measurements, and displaying data collected. The third section (4 chapters) is a basic introduction to probability and statistics. These chapters focus on basic statistical techniques; probability theory is given only a couple of pages.

The fourth and fifth sections are the most interesting, a total of 14 chapters on experimental design, analysis and simulation. Jain's skillful applications of these methods to problems in computer communication have made him widely respected in the field, and the chapters reflect his close knowledge of the subject.

Queuing Theory

The final section discusses queuing theory. This section is probably the weakest. Jain, with some good reasons, believes that queuing theory is not as useful as some of the other techniques presented in the book. But the presentation of queuing leaves out some useful techniques. In particular, M/G/1 systems, which are fairly congenial to solution, are not presented. The section also neglects Little's Law in Distribution, which eases the solution to many common types of queuing models [1, 2].

Impressive

Taken overall, the book is very impressive. The survey can be fairly called comprehensive, and the depth of discussion is generally appropriate for a survey. The text is usually a pleasure to read, a surprise for a highly mathematical book. Furthermore, the frequent use of case studies is effective and sometimes entertaining. (Indeed, I thought the case studies alone made interesting reading).

References

[1] Keilson J. & L. D. Servi, "A Distributional Form of Little's Law," Operations Research Letters, Vol. 7, No. 5, 1988, pp. 223–227

[2] Keilson J. & L. D. Servi, "The Distributional Form of Little's Law and the Fuhrmann-Cooper Decomposition," Operations Research Letters, Vol. 9, No. 4, 1990, pp. 239–247.

—Craig Partridge

[Ed.: Craig Partridge has recently joined the Swedish Institute of Computer Science (SICS) for a year of post-doctoral studies. We wish him all the best and look forward to his continuing contributions to the Internet in general and this journal in particular].

Network Operations discussion group

At INTEROP 90, a Birds-Of-a-Feather (BOF) session about network management surfaced a general desire for an online forum for discussing the realities of operating TCP/IP (and other) networks. Such a forum is pointedly different from technology-related discussions, such as the ones for the Simple Network Management Protocol (SNMP), which focus on one or more components or tools.

Joining the group

We have established a mailing list at `NET-OPS@DECWRL.DEC.COM`, to support this. To have yourself or a local redistributor added to the list, please send a note to `NET-OPS-REQUEST@DECWRL.DEC.COM`.

Focus

The list is unmoderated. It is intended for professional use, among those in the Internet attempting to operate actual open systems networks, such as those using TCP/IP, and those attempting to understand the process of operating such networks. The inclusion of "commercial content" is assumed to be inappropriate, unless the group discussion develops a desire to hear about product features.

—Dave Crocker, DEC Network Systems Lab.

DEC Technical Report on Congestion Management

The following DEC technical report is now available for external distribution: DEC-TR-724 *Myths About Congestion Management in High-Speed Networks* by Raj Jain.

Abstract

Weaknesses in several recently proposed ideas about congestion control and avoidance in high-speed networks are identified. Both sides of the debate concerning prior-reservation of resources versus walk-in service, open-loop control versus feedback control, rate control versus window control, and router-based control versus source-based control are presented. The circumstances under which backpressure is useful or not useful are discussed, and it is argued that a single congestion scheme is not sufficient, but that a combination of several schemes is required for complete congestion management in a network.

Getting the report

If you would like to receive a hard copy of this report, please send your address in a form suitable for direct application as a mailing label to the address below. (Any other words or sentences in the your letter/message will only delay the delivery).

Raj Jain
Digital Equipment Corporation
550 King St. LKG 1-2/A19
Littleton, MA 01460
`jain@erlang.enet.dec.com`

CONNEXIONS
480 San Antonio Road
Suite 100
Mountain View, CA 94040
415-941-3399
FAX: 415-949-1779

ADDRESS CORRECTION
REQUESTED

FIRST CLASS MAIL
U.S. POSTAGE
PAID
SAN JOSE, CA
PERMIT NO. 1

CONNEXIONS

EDITOR and PUBLISHER — Ole J. Jacobsen